Interpolation

September 23, 2015

There are many (topological) spaces of functions where polynomials are dense. For example, in the space of continuous functions with the *sup* norm; this is the Weierstrass approximation theorem.

Providing an "exact" value for a general real number means providing an infinite sequence of rationals — something that a computer cannot do in a finite amount of time. Providing an "exact" representation for a general function would require providing an uncountable collection of pairs of real numbers — which is (if anything!) "more impossible".

However, there are some (well-behaved) functions, for each of which we *can* write a program which can compute it to any prescribed level of accuracy. Unfortunately, such programs can be time consuming.

For these reasons (and others which we shall see by and by) it is useful to find good polynomial approximations to a function.

1 Formalism

Suppose we are given a finite collection $\xi = (x_0, \ldots, x_n)$ of x-values and the values $y_i = f(x_i)$ of the function at these points. The problem we now examine is to find a polynomial $P(x) = P_\xi(x)$ of degree n that takes *exactly* the same values at these points.

Writing $P(x) = \sum_{k=0}^{n} a_k x^k$, we see that the problem is that of solving the system of $n+1$ linear equations in $n+1$ variables a_i as follows:

$$a_0 + a_1 x_i + \cdots + a_n x_i^n = y_i \text{ for } i = 0, \ldots, n$$

The matrix version of this system of equations is:

$$\begin{pmatrix} 1 & x_0 & \ldots & x_0^n \\ 1 & x_1 & \ldots & x_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \ldots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

We know this matrix as the Vandermonde matrix whose determinant is the product of the terms $(x_i - x_j)$ (for $i < j$); which is non-zero since the x_i are assumed to be distinct. Hence, we *can* solve the equation to find the coefficients a_i *uniquely*. However, we would like to write a formula for the solution so that it can be calculated easily.

In the case $n = 1$, we can write the secant to the graph $y = f(x)$:

$$y - y_0 = \frac{y_1 - y_0}{x_1 - x_0}(x - x_0)$$

We see that the linear polynomial that interpolates the two values is given by:

$$P_{(x_1,x_0)}(x) = (\Delta^0_\xi f)_0 + (\Delta^1_\xi f)_0(x - x_0)$$

where we define $(\Delta^0_\xi f)_i = y_i$ and

$$(\Delta^1_\xi f)_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

How can one generalise this? Let us examine the solution for $n = 2$. We have

$$P(x) - P(x') = a_2(x^2 - x'^2) + a_1(x - x') = (a_2(x + x') + a_1)(x - x')$$

This gives

$$(\Delta^1_\xi P)_0 = a_2(x_1 + x_0) + a_1 \text{ and } (\Delta^1_\xi P)_1 = a_2(x_2 + x_1) + a_1$$

It follows that

$$a_2 = \frac{(\Delta^1_\xi f)_1 - (\Delta^1_\xi f)_0)}{x_2 - x_0}$$

This hints at the inductive definition (for $k \geq 0$)

$$(\Delta^{k+1}_\xi f)_i = \frac{(\Delta^k_\xi f)_{i+1} - (\Delta^k_\xi f)_i)}{x_{i+k} - x_i}$$

along with the expression

$$P_\xi(x) = \sum_{k=0}^{n} \left((\Delta^k_\xi f)_0 \prod_{i=0}^{k-1} (x - x_i) \right)$$

Consider the linear map from functions to $n+1$-tuples which is given by

$$f \mapsto \left((\Delta^0_\xi f)_0, (\Delta^1_\xi f)_0, \ldots (\Delta^n_\xi f)_0) \right)$$

We see that $\prod_{i=0}^{k-1}(x - x_i)$ goes to the standard basis vector e_k (with 1 in the k-th place and 0 elsewhere). This can be used to easily prove the above formulae for the polynomial interpolation of f.

2 Error term

We now ask ourselves how good this approximation is. To do so, we notice that $f(x) - P(x)$ vanishes at all the points x_i. So, if f were a polynomial of higher degree then $f(x) - P(x)$ would be a multiple of the polynomial $\prod_{i=0}^{n}(x - x_i)$. The size of this multiple is what will determine the accuracy of our approximation.

The polynomial $\prod_{i=0}^{n}(x - x_i)$ vanishes *precisely* at the points x_i. Thus, given any x' we can find $K_{x'}$ so that the following equality holds:

$$f(x') - P(x') = K_{x'} \prod_{i=0}(x' - x_i)$$

Since $K_{x'}$ is a measure of the error in the approximation of $f(x')$ by $P(x')$, we want to estimate it. (Obviously $K_{x'} = 0$ if $x' = x_i$ for some i, so we assume that $x' \neq x_i$ for any i).

To do so, we consider the function

$$E_{x'}(x) = f(x) - P(x) - K_{x'} \prod_{i=0}(x - x_i)$$

Let us assume that f is differentiable $n+1$ times in an interval that includes all the x_i and x'; then so is $E_{x'}$. Moreover, $E_{x'}$ has (at least) $n+2$ zeroes in this interval (at all the points x' and x_i for $i = 0, \ldots, n$). By Rolle's theorem, the derivative of $E_{x'}$ has $n+1$ zeroes, the second derivative n zeroes and so on. It follows that the $n+1$-st derivative has at least $n + 2 - (n+1) = 1$ zeroes. Let x'' be such a zero. Since the $n+1$-st derivative of $P(x)$ is 0 and the $n+1$-st derivative of $\prod_{i=0}^{n}(x - x_i)$ is $(n+1)!$, we have

$$f^{(n+1)}(x'') - K_{x'}(n+1)! = 0$$

In other words, we see that $K_{x'} = f(x'')/(n+1)!$ for some x'' in the smallest interval $[a, b]$ which contains x' and all the x_i for $i = 0, \ldots, n$.

If we additionally assume that f is $n+1$-times *continuously* differentiable, then $f^{(n+1)}$ is continuous and thus uniformly bounded in this interval $[a, b]$ by some constant $M(a, b)$. It follows that the error in the approximation is then bounded as follows

$$|f(x) - P_\xi(x)| \leq M(a, b) \left| \prod_{i=0}^{n}(x - x_i) \right|$$

In particular, if we choose x_i to be closely spaced and x to lie in the interval that contains the x_i, then we can make the approximation good.

3 Computation

In many algebraic situations, we are satisfied when we find a closed form formula for something. When the terms of such a formula are difficult to calculate accurately, it may sometimes be better to go back to first principles to see how the formula can be derived differently in order to make a more accurate and efficient algorithm. The calculation of $P(x)$ is one such case.

Calculation of divided differences involves division by numbers of the form $(x_i - x_j)$ which may be quite small. It is thus convenient to organise the calculation in a way that retains as much accuracy as possible.

As before, we assume that we are given $\xi = (x_0, \ldots, x_n)$. Now suppose that g_1 and g_2 are functions so that

$$g_1(x_{i_1}) = g_2(x_{i_1}); \ldots; g_1(x_{i_k}) = g_2(x_{i_k})$$

Let us define

$$h(x) = \frac{1}{x_{i_{k+1}} - x_{i_0}} \det \begin{pmatrix} g_2(x) & x - x_{i_{k+1}} \\ g_1(x) & x - x_{i_0} \end{pmatrix}$$

We calculate that $h(x_{i_j}) = g_1(x_{i_j})$ for $j = 1, \ldots, k$. Moreover, we see that

$$h(x_{i_0}) = g_1(x_{i_0}) \text{ and } h(x_{i_{k+1}}) = g_2(x_{i_{k+1}})$$

The above calculation allows us to define interpolation polynomials $P_{i+1,\ldots,i+k}(x)$ as follows. We define $P_i(x) = y_i$ (constant function). We then inductively define

$$P_{i,i+1,\ldots,i+k}(x) = \frac{1}{x_{i+k} - x_i} \det \begin{pmatrix} P_{i+1,\ldots,i+k}(x) & x - x_{i_k} \\ P_{i,\ldots,i+(k-1)}(x) & x - x_i \end{pmatrix}$$

By induction, we can show that $P_{i,\ldots,i+k}(x_{i+r}) = y_{i+r}$ for $r = 0, \ldots, k$. In particular, we see that $P_{0,\ldots,n}(x)$ is the interpolation polynomial that we are looking for.

Now, this is a very cumbersome way of writing the polynomial if we want to write it *algebraically*. However, it is a very efficient way of *calculating* the value of $P(x)$ at a *given* value of x. The reason is that at each stage we are only calculating a 2x2 determinant; this involves only two multiplications.

In order to make the calculation more accurate, we can also re-order the calculation as follows. We note that the polynomial $P_\xi(x)$ is unique and does not depend on the numbering of the x_i. Thus it is possible to number the points x_i so that $|x - x_i|$ is a non-decreasing sequence. Such an organisation apparently results in greater accuracy.